Amendments to the Specification:

Please replace the paragraphs indicated below for the paragraphs previously presented. Changes are shown with additions double underlined and deletions in ~~strikethrough text~~. No new matter is added by these amendments.

The paragraph under the heading Cross-Reference to Related Applications:

This application is a continuation of U.S. Patent Application Serial No. 09/432,362, filed November 3, 1999, which claims the benefit of priority to Provisional Application Serial No. 60/106,904, filed 3 November 1998, each of which are~~is~~ hereby incorporated by reference. ~~This application also claims the benefit of priority to~~ Application Serial No. 09/432,362 is a continuation-in-part of U.S. Patent Application Serial No. 09/076,617 filed 12 May 1998, now U.S. Patent No. 6,042,555, which itself incorporates by reference and claims the benefit of priority to U.S. Provisional Application Serial No. 60/046,185, filed 12 May 1997, each of which are hereby incorporated by reference.

The paragraph beginning on page 11, line 13:

Figure 1 shows a diagrammatic view of the relationship between a *Host Simulation Application* (101) and Virtual Technologies' VirtualHand Toolkit (VHT), a portion of which is the subject of this invention. Figure 1 also shows the relationship between the VHT and a *Hand Manager* module capable of constraining the virtual hand from penetrating graphical objects. In particular, we describe a *Hand-Interaction Scene Graph* (or more generally, the *Interaction Scene Graph*), which functions within a *Hand-Interaction Simulation*. This hand-interaction scene graph interacts with a simulation's *Master (a.k.a. World or Graphical) Scene Graph.*

The paragraph beginning on page 11, line 25 through the paragraph ending on page 12, line 13:

The VirtualHand Toolkit, which makes up a large portion of the supporting software, comprises the following three main components:

1. Hand-Interaction Simulation (H.I.S.) (104)

2. I/O Daemon (106)

3. Calibration Panel (107)

Hand-interaction Simulation (HIS)

A central module of the VirtualHand Toolkit architecture is the *Hand-Interaction Simulator (HIS)* 104, which controls all hand interaction. It may be a separate process running on the host computer or, alternatively, if the Virtual Technologies CyberGrasp haptic-feedback system is used with the CyberGlove, the HIS may alternatively run on the CyberGrasp controller. The HIS comprises four major components:

1. Hand-Interaction Scene Graph (103)(or more generally, just Interaction Scene Graph)

2. Collision Manager (108)

3. State Manager (109)

4. Hand Manager (102)

The paragraph beginning on page 13, line 1 through the paragraph ending on page 14, line 4:

The HIS may also contain a *State Manager* (109)which is responsible for updating the state of each of the nodes that are in the scene graph. During each cycle through the HIS, the state manager traverses the HISG tree, assigning new states (e.g., positions, velocities, etc.) to the various dynamic nodes. If the HIS is not a process running on the host computer, typically, the state manager updates the states of the HISG running in the HIS only, and not the HISG in

the host application. In such a case, typically only the HISG in the HIS will have the ability to synchronize with the corresponding HISG in the host application.

A *Collision Manager* (108) navigates the HISG and determines which dynamic nodes are colliding with one another. Static nodes are typically ignored by the collision manager. Support may include sphere/plane, sphere/box and sphere/sphere collisions, as well as other object collisions. The hand itself is part of the haptic scene graph and consists of multiple dynamic nodes. In one embodiment, we use multiple dynamic sphere nodes (i.e., contact *spheres*) in the fingers and hand to check for collision with other objects, while other embodiments may use multiple dynamic bounding cylinders on the phalanges of the fingers, and a bounding box on the palm, as an alternatives or in addition to the spheres. The collision manager has at least two objectives: to determine which two nodes (objects) are colliding, and to determine how deeply inside one another they are (i.e., the depth of penetration). The depth of penetration may be used to compute force feedback commands based on a stiffness model.

It should be noted that the nodes in the hand-interaction scene graph do not necessarily correspond exactly to the nodes in a graphics (e.g., the master) scene graph (105). For example, a complex shape could be rendered "as is" graphically, but from a haptic point of view, it would be represented by a bounding box.

In another embodiment, there is another *Data Neutral Scene Graph* (DNSG) for coordinating and linking data between the master (i.e., graphical) scene graph 105 and the HISG. The DNSG contains a node for each interactive element or other element of interest existing in either the graphical scene graph or the HISG. Elements of interest typically include elements which might change with time. When a DNSG is employed, the DNSG is typically the scene graph which is updated with each pass through the simulation loop. (That is, in effect, it becomes

the "master" scene graph.) After the DNSG is updated, it passes the update information along to the corresponding node of the graphical scene graph and/or the HISG for updating. When a DNSG is used, the HISG typically sends force commands, or local force patch approximations, to an I/O Daemon (106) (discussed below) running on a force controller.

The paragraph starting on page 15, line 11 through the paragraph ending on page 17, line 21:

Host Application

The VirtualHand Toolkit (VHT) may be used with a high-level software simulation package. Common commercial package examples include Sense8's World Toolkit™, Division's dView™, etc. In such cases, these applications (101) already have defined a graphical master (i.e., world) scene graph (105), and data must be extracted from the master scene graph to construct a hand-interaction scene graph (110). (Note also that a DNSG as previously described might preferably be constructed to efficiently communicate with the graphical scene graph and the HISG.) Hence, a custom plug-in (112) to the host application is required to permit the master scene graph to communicate with a host copy of the HISG. Such a HISG plug-in is shown in the host application and represents a host-application copy of the HISG. The master scene graph and the host application HISG copy may synchronize each other using the DNSG. They may also synchronize with each other using either of the following two communication methods: *poll* mode or *call-back* mode. In poll mode, the master scene graph requests the new data from the host application HISG and updates the master states itself. In call-back mode, the master scene graph passes a function pointer to the host HISG copy which then directly updates the object states in the master scene graph. When the HIS is running on a separate computer, such as a

force controller, then a "mediator" process running in the HIS let the HISGs running in the host application and in the HIS update one another asynchronously.

Most commonly, when using the toolkit, the user will modify the *Application* software such that a hand-interaction scene graph may be constructed from, and/or communicate with, the master scene graph (or alternatively, the DNSG). Typically, applications using the VirtualHand Toolkit fall within three scenarios:

SCENARIO #1

The VHT is being used with a third party software package such as the world-building toolkits provided by Sense8 Corp. Division Inc. and Deneb Robotics Inc. In this case, the applications already have a master scene graph (105) and data must be extracted from it to construct a hand-interaction scene graph (110)(and/or a DNSG (111)) residing on the host. If the HIS is running on a separate computer, such as a force controller, then this hand-interaction scene graph may be copied over to the HIS and the two are updated asynchronously. This is done because the simulation running on the host will typically run at a much slower rate than the simulation running on the dedicated CyberGrasp controller. A high servo rate is desirable for high-fidelity feedback.

SCENARIO #2

In this scenario, an application (114) that is "interaction scene graph aware" is being written from the ground up using OpenGL, OpenInventor, etc. In this case, the interaction scene graph (113) may be directly incorporated into the master scene graph.

SCENARIO #3

In some instances, the users might want to bypass the interaction scene graph altogether and write their own hand interaction simulation (116). In these instances, they can communicate directly with the I/O daemon to read data from the devices and control them directly via a connection (115).

Embodiment

An embodiment which illustrates the functionality of the VirtualHand Toolkit consists of a simulation of a virtual control panel (201) (Figure 2). The user (200) may interact with the control panel and buttons using a CyberGlove instrumented glove, and optionally with a CyberGrasp force-feedback device. The control panel contains grounded objects (202) and surfaces (204), in addition to push buttons and toggle switches (203). When the CyberGrasp force-feedback device is used, the user can also feel the resistance of the buttons being pressed.

Figure 4 is a link-revolute-joint model of the human hand (400). The forearm link (401) is connected to the wrist yaw joint (402) which is connected to the wrist pitch joint (403). This joint is connected to the metacarpus link (404) which extends to connect to the thumb roll joint (405). This joint is connected to the thumb abduction joint (406) which is connected by the thumb metacarpal link (407) to the thumb metacarpal roll joint (408). This joint is connected to the thumb MP joint (409) which is connected by the thumb proximal phalanx (410) to the thumb IP joint (411), which is connected to the thumb distal phalanx (412). The hand metacarpus link (404) also connect to the joints at the base of each of the fingers (420). It connects to the index abduction joint (413), which connects to the index MP joint (414). The index MP joint is connected by the index proximal phalanx link (415) to the index 15PI joint (416) which is connected by the index medial phalanx link (417) to the index DI joint (418) which ultimately

connects to the index distal phalanx link (419). The structure is similar for each of the middle, ring and small fingers. Such a link-revolute joint hand model is described in Kramer's U.S. Patent No. 5,482,056, "Determination of Thumb Position Using Measurements of Abduction and Rotation." Link-revolute joint models for the entire human body are described in Kramer's U.S. Patent No. 5,676,157, "Determination of Kinematically Constrained Multi-Articulated Structures."